ROBUST BICLUSTERING ALGORITHM *(ROBA)* FOR DNA MICROARRAY DATA ANALYSIS

Alain B. Tchagang *tcha0003@umn.edu*
Ahmed H. Tewfik *tewfik@umn.edu*

Electrical and Computer Engineering
University of Minnesota, 200 Union St. SE Minneapolis, MN 55455, USA

ABSTRACT

Recently, biclustering algorithms have been used to extract useful information from large sets of *DNA* microarray experimental data. They refer to a distinct class of clustering algorithms that perform simultaneous row-column clustering. The goal is to find submatrices, that is, subgroups of genes and subgroups of conditions, where the genes exhibit highly correlated activities for every condition. Almost all of the methods proposed in the literature search for one or two types of bicluster among four. Also, most of the proposed methods rely on solving an optimization problem. Therefore, the method is dependant on the optimally criterion which most of the time, is likely to miss some significant biclusters. In this study, we develop a Robust Biclustering Algorithm (*ROBA*) to address some of the issues mentioned above. Our algorithm is simple because it uses basic linear algebra and arithmetic tools and there is no need to solve and optimization problem. Our algorithm is robust because it can be used to search for any type of bicluster defined by the user in a timely manner and, it is also shown to be more efficient than the ones proposed in the literature.

1. INTRODUCTION

One of the major goals of gene expression data analysis is to uncover genetic pathways. This task is difficult because subgroups of genes display similar activation patterns *only* under certain experimental conditions. Genes that are coregulated or coexpressed under a subset of conditions will behave differently under other conditions. Finding genetic pathways therefore requires identifying clusters of genes that are coexpressed under subsets of conditions as opposed to all conditions. Gene expression data is typically arranged in a data matrix, with rows corresponding to genes and columns to experimental conditions. The $(n, m)^{th}$ entry of the gene expression matrix represents the expression level of the gene corresponding to row n under the specific condition corresponding to column m. Finding the genetic pathways is therefore equivalent to simultaneously clustering the rows and columns of the gene expression matrix. Cheng and Church [1] introduced the term biclustering to denote simultaneous row-column clustering of gene expression data. It should be clear that biclustering techniques produce local models whereas clustering approaches compute global models. If we use a clustering algorithm on the rows of the gene expression matrix, a given gene cluster is defined using all the conditions. In contrast, a biclustering technique will assign a gene to a bicluster based on a subset of conditions.

In the literature, there exist many biclustering algorithms that have been developed [2-9]. Most of those previous techniques search for one or two types of biclusters among four that have been identified in the literature [1]: biclusters with constant values, biclusters with constant values on rows or columns, biclusters with coherent values, and biclusters with coherent evolution. Also, most of the proposed methods rely on solving an optimization problem. Therefore, the method is dependant on the optimally criterion which most of the time, is likely to miss some significant biclusters. In this study, we develop a Robust Biclustering Algorithm (*ROBA*) to address some of the issues mentioned above. Our algorithm is simple because it uses basic linear algebra and arithmetic tools and there is no need to solve and optimization problem. Our algorithm is robust because it can be used to search for any type of bicluster defined by the user in a timely manner and, it is also shown to be more efficient than the ones proposed in the literature. We illustrate our algorithm here by focusing on the identification of biclusters with constant values, biclusters with constant values on rows or columns, and biclusters with coherent values. The rest of this paper is organized as follows. After a quick description of gene expression matrix in section *2*, we develop part of the Robust Biclustering Algorithm in section *3*. In section 4, we show some simulation results and we compare the proposed biclustering algorithm with previous ones.

2. GENE EXPRESSION MATRIX

A *DNA* microarray data can be represented as an *NxM* matrix *A* whose rows represent the genes, columns the

experimental conditions, and real number entries a_{nm} the expression level of gene *n* under condition *m* as illustrated in equation (1)

$$A = \begin{bmatrix} a_{11} & a_{12} & \dots & a_{1M} \\ a_{21} & a_{22} & \dots & a_{2M} \\ \vdots & \vdots & \vdots & \vdots \\ a_{n1} & a_{n2} & \dots & a_{nM} \\ \vdots & \vdots & \vdots & \vdots \\ a_{N1} & a_{N2} & \dots & a_{NM} \end{bmatrix}. \quad (1)$$

We can also partition the matrix *A* into rows, or into columns as illustrates by equation 2 and equation 3 respectively.

$$A = \begin{bmatrix} R_1 & R_2 & \cdots & R_n & \cdots & R_N \end{bmatrix}^T \quad (2)$$

$$A = \begin{bmatrix} C_1 & C_2 & \cdots & C_m & \cdots & C_M \end{bmatrix} \quad (3)$$

In 2 and 3, $R_n = \begin{bmatrix} a_{n1} & a_{n2} & \cdots & a_{nm} & \cdots & a_{nM} \end{bmatrix}$, $C_m = \begin{bmatrix} a_{1m} & a_{2m} & \cdots & a_{mm} & \cdots & a_{Mm} \end{bmatrix}^T$, with *n = 1* to *N*, and *m = 1* to *M*. The row vector R_n corresponds to the expression levels of the n^{th} gene under *M* conditions. The column vector C_m corresponds to the expression levels of the *N* genes under the m^{th} condition. From equation 1, we can also define two additional vectors: the row vector *Conditions (1xM)* and the column vector *Genes (Nx1)*. They are both label vectors and they are defined to keep track of every condition and gene.

Conditions = [Condition1 ... Condition m ... Condition M]

Genes = [Gene1 Gene2 Gene 3 ... Gene n ... Gene N]^T

3. THE ROBUST BICLUSTERING ALGORITHM

The Robust Biclustering Algorithm works as follows. After solving the problems of missing values noise corruption using any of the known techniques or a simple approach that we describe below, the gene expression matrix is written as the sum of the product of each of its distinct elements with an elementary matrix. Each elementary matrix is binary, i.e., its elements are either "1" or "0". By performing elementary row or the column operations on the elementary matrices, it becomes easy to identify all perfect biclusters in a timely manner.

3.1. Data Conditioning

The first part of the proposed biclustering algorithm consists of performing the data conditioning due to the fact that we are not only working with noisy data but also the *DNA* experimental data contains missing values. One has to note that, this step is not necessary. It is performed only if the data to analyze contains missing values or noise. Many techniques to recover missing values have been developed in the literature [10, 11]. Since the recovery of missing values is not our main focus in this study, we have used the zero method, i.e., replacing each missing value by zero. Several techniques have been proposed in the literature, to deal with noise, including many data quantization techniques. In this study we have used the following approach. First, we identify the number *L* of distinct values α_l that exists in the gene expression matrix *A*. We assume that the values α_l are rank ordered according to their magnitudes, i.e., $\alpha_l < \alpha_{l+1}$ .Next, we redefine α_l using equation 4:

$$\alpha_l = (b_l + b_{l-1})/2 \quad (4)$$

Where: $b_l = b_0 + le$, with $l = 1$ *to* L, $e = (b_L - b_0)/L$, $b_0 = min([a_{nm}])$, and $b_L = max([a_{nm}])$. The interval $[b_0\ b_L]$ is then divided into *L* equal intervals :

$[b_0\ b_L] = [b_0\ b_1[\ U \dots U\ [b_{L-1}\ b_l[\ U \dots U\ [b_{L-1}\ b_L]$.

Finally, a new data matrix is obtained using Algorithm1. One advantage of using this quantization approach is that it does operate on all the data of the matrix. Therefore the biclusters that are present in the original set of data are not likely to be destroyed. All it does is to reduce the number of original biclusters and increase their size by merging some of them together.

Algorithm 1

Input A = Microarray Data
Output A = Quantized Microarray Data
Begin,
Compute: L, b_L, b_0, e, b_l, α_l
For l = 1 to L
  For n = 1 to N
    For m = 1 to M
      If $a_{nm} \in [b_{l-1}\ b_l[$
        $a_{nm} = \alpha_l$
      elseif $a_{nm} == b_L$
        $a_{nm} = \alpha_L$
      End
    End
  End
End
End Begin.

3.2. Gene Expression Matrix Decomposition

The second part of the proposed biclustering algorithm consists of writing the matrix *A* as the sum of the product of each of its distinct elements with an elementary matrix. It is the first important step of the proposed biclustering algorithm because, after the gene expression matrix is written as mentioned above, obtaining perfect biclusters is straightforward. This is due to the fact that we are just dealing with two numbers: "0" and "1".

Given that A is made up of L distinct values, A can be expressed using equation 5.

$$A = \sum_{l=1}^{l=L} \alpha_l A_l = \alpha_1 A_1 + \ldots + \alpha_L A_L \quad (5)$$

From equation 5, we observe that the A_l's are binary matrices as we mentioned earlier. As above, we can also partition them as rows or columns as illustrated by equation 6 and 7 respectively.

$$A_l = \begin{bmatrix} r_1^l & r_2^l & \cdots & r_n^l & \cdots & r_N^l \end{bmatrix}^T \quad (6)$$

$$A_l = \begin{bmatrix} c_1^l & c_2^l & \cdots & c_m^l & \cdots & c_M^l \end{bmatrix}^T \quad (7)$$

Also, in equations 6 and 7 respectively, the rows vectors r_n^l are binary *1xM* vectors and the columns vectors c_m^l 's are binary *Nx1* vectors. The row vector r_n^l corresponds to the n^{th} row of the elementary matrix that is associated to the l^{th} distinct element of the gene expression matrix. The column vector c_m^l corresponds to the m^{th} column of the elementary matrix that is associated to the l^{th} distinct element of the gene expression matrix. From equations 2, 3, 4, 5, 6, and 7 we can derive the following relations.

$$R_n = \sum_{l=1}^{l=L} \alpha_l r_n^l \,,\; C_m = \sum_{l=1}^{l=L} \alpha_l c_m^l \,,\; \sum_{l=1}^{l=L} A_l = ones(N,M).$$

Decomposing the gene expression matrix as shown above has many advantages. The first one, as mentioned earlier, is that it allows the algorithm to operate on binary data. Thus we gain in terms of computational complexity and memory resources. Secondly, it allows the user to get more local information about the gene expression matrix in a simple way. For example the ones in the binary row vector r_n^l show the positions (that is the conditions) at which the n^{th} gene has the same expression value α_l (which corresponds to the l^{th} distinct element of the gene expression matrix) and its zeros show the position at which the same n^{th} gene is not expressed at α_l. On the other hand, the ones in the binary column vector c_m^l show subgroups of genes that have the same expression value α_l under the same m^{th} condition, and its zeros show the subgroup of genes that are not expressed at the same value α_l under the same m^{th} condition. Therefore this observation is very useful. For example, if one is given two genes with two different binary rows vectors: r_n^l and r_k^l associated with the same expression values α_l, one can identify the position at which both genes are expressed simultaneously at α_l by performing element wise product r_n^l and r_k^l. The result will be a binary row vector with its ones showing the positions at which both genes are expressed simultaneously at α_l. As will become clear below, this observation also plays a critical role in the elaboration of the Robust Biclustering Algorithm. Finally, observe that the decomposition is also a powerful gene expression visualization tool.

3.3. Biclusters Identification

The third part of the proposed algorithm consists of identifying the four types of biclusters from the gene expression matrix. First we develop three simple algorithms that can be used to extract all biclusters with constant values, biclusters with constant values on columns, biclusters with constant values on rows. Secondly, we show how by adding few operations to one of them, the modified algorithm can be used to extract biclusters with coherent values.

3.3.1. Biclusters with Constant Values

In a *DNA* microarray experimental data, a perfect bicluster with constant values is any submatrix *B (IxJ)* of *A* whose elements are constant:

$$B = [a_{ij}] = \mu.Ones(I,J) \quad (8)$$

with: $a_{ij} = \mu$, $i = 1$ to I, $j = 1$ to J. Such matrices reveal subgroups of genes with constant expression levels within a subgroup of conditions or vice versa.

From the gene expression matrix decomposition performed above, such matrices can be obtained by analyzing each elementary matrix A_l separately to obtain subgroups of genes that have constant expression level α_l under different conditions. Since A_l is a binary matrix, and since the number of genes N is always greater than the number of conditions M, the number of biclusters (N_b) with constant values in a *DNA* microarray experimental data can be defined using equation (9).

$$N_b = \sum_{l=1}^{l=L} P_l \quad (9)$$

P_l is the number of distinct non zeros rows r_i^l of each elementary matrix A_l. Now note that each distinct non zeros row r_i^l of each elementary matrix A_l constitutes the principal row element of the i^{th} bicluster B_i^l of the elementary matrix A_l considered. Therefore, in order for any other row r_n^l of the elementary matrix A_l to belong to the i^{th} bicluster, equation (10) has to be true, that is the element wise product of the two given row vectors.

$$r_i^l .* r_n^l = r_i^l \quad (10)$$

with: $i = 1$ to P_l, $n = 1$ to N, and $l = 1$ to L. Algorithm 2 is then used to extract biclusters that have constant expression level α_l.

Algorithm 2

Input: A = Quantized Microarray Data
Output: B_i^l *= Biclusters with Constant Values*
Begin,
Compute: P_l, r_i^l,r_n^l
For l = 1 to L
For i = 1 to P_l
$B_i^l = []$;
For n = 1 to N
If $r_i^l .* r_n^l == r_i^l$
$B_i^l = [B_i^l$; *[Genes(n)* $\alpha_l r_i^l]]$
End
End
End
End; $B_i^l = [[0$ *Conditions];* $B_i^l]$;
End Begin.

3.3.2 Biclusters with Constant Values on Columns

A perfect bicluster with constant values on a column is any submatrix *B (IxJ)* of *A* which has one of the following forms:

$$B = \left[a_{ij} \right] = \begin{cases} \left[\mu + \beta_j \right], \text{additive model,} \\ \left[\mu\beta_j \right], \text{multiplicative model.} \end{cases}$$

The general form can be represented using equation (11).

$$B = \begin{bmatrix} . & . & \cdots & . \\ \mu_1 & \mu_2 & \cdots & \mu_J \\ . & . & \cdots & . \end{bmatrix} \quad (11)$$

In a *DNA* microarray experimental data, biclusters with constant values on columns identify subgroups of conditions within which a subgroup of genes present similar expression values assuming that the expression values may differ from condition to condition. From the gene expression matrix decomposition performed above, the number of biclusters *(*N_b*)* with constant values on columns is given by equation (12).

$$N_b = P_c \quad (12)$$

P_c is the number of distinct non zeros columns c_j of the entire elementary matrices A_l. Once more, each distinct column c_j of the entire elementary matrices A_l constitutes the principal column element of the j^{th} bicluster B_j .Therefore, in order for any other column c_m^l of any elementary matrix A_l to belong to the j^{th} bicluster, equation (13) has to be verified: that is the element wise product of the two given column vectors.

$$c_j .* c_m^l = c_j \quad (13)$$

with: *j = 1* to P_c, *m = 1* to *M*, and *l = 1* to *L*. Algorithm 3 is then used to extract biclusters that have constant values on columns.

Algorithm 3

Input: A = Quantized Microarray Data
Output: B_j *= Biclusters with Constant Values on Columns*
Begin,
Compute: P_c, c_j, c_m^l
For j = 1 to P_c
$B_j = []$;
For l = 1 to L
For m = 1 to M
If $c_j .* c_m^l == c_j$
$B_j = [B_j$ *[Conditions(m) ;* $\alpha_l c_j]]$
End
End
End; $B_j = [[0$ *Genes]* $B_j]$;
End
End Begin.

3.3.3. Biclusters with Constant Values on Rows

A perfect bicluster with constant values on rows is any submatrix *B (IxJ)* of *A* which has one of the following forms:

$$B = \left[a_{ij} \right] = \begin{cases} \left[\mu + \alpha_i \right], \text{additive model,} \\ \left[\mu\alpha_i \right], \text{multiplicative model.} \end{cases}$$

The general form can be represented using equation (14).

$$B = \begin{bmatrix} \cdots & \mu_1 & \cdots \\ \cdots & \mu_2 & \cdots \\ \cdots & \cdots & \cdots \\ \cdots & \mu_I & \cdots \end{bmatrix} \quad (14)$$

In a *DNA* microarray experimental data, biclusters with constant values on rows represent subgroups of genes with similar expression level across a subgroup of conditions, allowing the expression levels to differ from gene to gene. From the gene expression matrix decomposition performed above, the number of biclusters *(*N_b*)* with constant values on rows is given by equation (15).

$$N_b = P_r \quad (15)$$

where P_r is the number of distinct non zeros rows r_i of the entire elementary matrices A_l. Each distinct row r_i of the entire elementary matrices A_l constitutes the principal row element of the i^{th} bicluster B_i. Therefore, in order for

any other row r_n^l to belong to the i^{th} bicluster, equation (16) has to be verified: that is the element wise product of the two given row vectors.

$$r_i \;.*\; r_n^l = r_i \qquad (16)$$

with $i = 1$ to P_r, $n = 1$ to N, and $l = 1$ to L. Algorithm 4 is then used to extract biclusters that have constant value on rows.

Algorithm 4

Input: A = Quantized Microarray Data
Output: B_i= Biclusters with Constant Values on Rows
Begin,
Compute: P_r, r_i, r_n^l
For i = 1 to P_r

$B_i = [];$

For l = 1 to L
For n = 1 to N
If $r_i \;.*\; r_n^l == r_i$
B_i = [B_i ; [Genes(n) α_l r_i]]
End
End
End; B_i = [[0 Conditions]; B_i];
End
End Begin.

3.3.4. Biclusters with Coherent Values

A perfect bicluster with coherent values is any submatrix *B (IxJ)* of *A* which has one of the following forms.
$B = [a_{ij}]$, with $a_{ij} = \mu + \alpha_i + \beta_j$ additive model or $a_{ij} = \mu.\alpha_i.\beta_j$, multiplicative model. In this study, we will only deal with additive model.
In a DNA microarray experimental data, biclusters with coherent values represent subgroups of genes and subgroups of conditions with coherent values on both rows and columns. $B = [\mu + \alpha_i + \beta_j] = [\mu] + [\alpha_i] + [\beta_j]$ can be viewed as the sum of three matrices: B_1 with constant values, B_2 with constant values on rows, and B_3 with constant values on columns. Therefore, to obtain perfect biclusters with coherent values from a *DNA* microarray experimental data, the following approach can be used.

Approach: The Gene expression matrix *A* is first written as the sum of three matrices Z_1 , Z_2, and Z_3 where Z_1 is a matrix with constant values, Z_2 a matrix with constant values on columns and $Z_3 = A - (Z_1 + Z_2)$. Next, use algorithm *4* to extract all perfect biclusters with constant values on rows from Z_3. Next, add them back to their corresponding matches into Z_1 and Z_2 and finally, obtain subgroups of gene with coherent values.

The choice of the matrix $Z_1 + Z_2$ which has constant values on columns is not arbitrary. It must be constructed using each row of the gene expression matrix *A* that is also part of the bicluster with coherent values see the bellow property.

Property: *Let X be a matrix that contains a bicluster with coherent values embedded within its structure. By subtracting from X a matrix Y that has constant values on columns, and which is constructed using a row of X that is also part of the bicluster with coherent values, the result is a matrix Z that contains a bicluster with constant values on rows embedded within its structure and located at the same address as the bicluster with coherent values.*

Proof: Without loss of generality, consider a matrix *X* that includes a bicluster with coherent values embedded in it:

$$X = \begin{bmatrix} a & \alpha_1+\beta_2 & f & \alpha_1+\beta_4 & \alpha_1+\beta_5 \\ b & e & g & j & k \\ c & \alpha_3+\beta_2 & h & \alpha_3+\beta_4 & \alpha_3+\beta_5 \\ d & \alpha_4+\beta_2 & i & \alpha_4+\beta_4 & \alpha_4+\beta_5 \end{bmatrix}.$$

The bicluster with coherent values $B = (\alpha_i + \beta_j)$ embedded within the structure of *X* is:

$$B = \begin{bmatrix} .. & \alpha_1+\beta_2 & .. & \alpha_1+\beta_4 & \alpha_1+\beta_5 \\ .. & .. & .. & .. & .. \\ .. & \alpha_3+\beta_2 & .. & \alpha_3+\beta_4 & \alpha_3+\beta_5 \\ .. & \alpha_4+\beta_2 & .. & \alpha_4+\beta_4 & \alpha_4+\beta_5 \end{bmatrix}$$

Thus we can construct the matrix *Y* that has constant values on columns using either the first, third or fourth row of *X*. Let's use the first row of *X*. Therefore, we have:

$$Y = \begin{bmatrix} a & \alpha_1+\beta_2 & f & \alpha_1+\beta_4 & \alpha_1+\beta_5 \\ a & \alpha_1+\beta_2 & f & \alpha_1+\beta_4 & \alpha_1+\beta_5 \\ a & \alpha_1+\beta_2 & f & \alpha_1+\beta_4 & \alpha_1+\beta_5 \\ a & \alpha_1+\beta_2 & f & \alpha_1+\beta_4 & \alpha_1+\beta_5 \end{bmatrix}$$

By computing Z = *X* – *Y*, we have:.

$$Z = \begin{bmatrix} 0 & 0 & 0 & 0 & 0 \\ b-a & e-\alpha_1-\beta_2 & g-f & j-\alpha_1-\beta_4 & k-\alpha_1-\beta_5 \\ c-a & \alpha_3-\alpha_1 & h-f & \alpha_3-\alpha_1 & \alpha_3-\alpha_1 \\ d-a & \alpha_4-\alpha_1 & i-f & \alpha_4-\alpha_1 & \alpha_4-\alpha_1 \end{bmatrix}$$

From the expression of *Z* obtained above, we can observe that *Z* has a bicluster with constant values on rows *Bc* embedded within the structure of *Z* and located at the same address as the bicluster with coherent values *B* embedded within the structure of the matrix *X*.

$$Bc = \begin{bmatrix} .. & 0 & .. & 0 & 0 \\ .. & .. & .. & .. & .. \\ .. & \alpha_3-\alpha_1 & .. & \alpha_3-\alpha_1 & \alpha_3-\alpha_1 \\ .. & \alpha_4-\alpha_1 & .. & \alpha_4-\alpha_1 & \alpha_4-\alpha_1 \end{bmatrix}$$

Since we do not have any knowledge about the rows of the gene expression matrix *A*, the intuitive approach is to use an iterative multi step approach. First, we iteratively

construct the matrix $Z_1 + Z_2$ which has constant values on columns using each row of A. Secondly, after each construction, we obtain $Z_3 = A - (Z_1 + Z_2)$. Next, we use algorithm *4* to extract all perfect biclusters with constant values on rows from Z_3. Finally, we add these biclusters back to their corresponding matches into $(Z_1 + Z_2)$ and obtain biclusters with coherent values. From the proof of the above property, we observe that there are many ways to construct the matrix $Z_1 + Z_2$ with constant values on columns and obtain the same bicluster with coherent value. Therefore, to avoid redundancy and gain in computational time, a strategy that allows the algorithm not to obtain a bicluster more than once must be defined.

4. RESULTS AND CONCLUSION

Let us conclude by discussing some of the results that we have obtained. As in [12], we have implemented the Robust Biclustering Algorithm in Matlab and tested it on the yeast gene microarray data that can be found at [13]. The data consists of *2884* genes and *17* conditions. We have obtained the following results. Initially, the data contained: $L = 206$ distinct values. We set $b_L = max[a_{nm}] = 595$, $b_0 = min[a_{nm}] = 0$ thus $e = 2.8883$, and $b_l = b_0 + le = 2.8883l$, with $l = 1$ *to* L. After data conditioning, we obtained $L = 111$ new distinct values. Then from our simulation, we obtained $N_b = 10225$ biclusters with constant values, $N_b = 3391$ biclusters with constant values on rows, and $N_b = 836$ biclusters with constant values on columns. Because of the large number of biclusters found, we will present here a few illustrative results. Figure 1 shows an example of biclusters with constant values, biclusters with constant values on rows and biclusters with constant values on columns obtained. Figure 2 shows an example of biclusters with coherent values obtained.

Finally, the Robust Biclustering Algorithm can be shown to have complexity of $O(N\text{x}M\text{x}L\text{x}N_b)$, where N is the number of rows of the gene expression matrix A, M is the number of column in A, L the number of distinct values in A, and N_b the number of biclusters. Thus it is less complex than the *FLOC* algorithm proposed by Yang et al which has complexity $O((N+M)^2\text{x}K\text{x}P)$, where P is the desired number of biclusters and K the number of iteration till the end. *FLOC* was shown by Yang et al. to be less complex than the Cheng and Church algorithm [9]. To quantify the complexity, after data conditioning and decomposition which take approximately *250s*, it takes less than *10s* to *ROBA* to get a bicluster. Thus its running time is better than that of [2] which reportedly takes *300-400s* to find a single bicluster. In our future work, we will be looking into the biological meanings of the huge amount of biclusters obtained.

5. REFERENCES

[1]- S. C. Madeira, A. L. Oliveira, "Biclustering Algorithms for Biological Data Analysis: A Survey", IEEE Transactions on computational Biology and Bioinformatics, Vol. 1, No. 1, Jan-March 2004.

[2]- Y. Cheng, G.M. Church, "Biclustering of Expression Data", In Proc. ISMB'00, pages 93-103. AAAI Press, 2000.

[3]- G. Getz, E. Levine, E. Domany, "Coupled Two-way Clustering Analysis of Microarray Data, Proc. Natl. Acad. Sci. USA, 97(22): 12079-84, 2000.

[4]- S. Bergmann, J. Ihmels, N. Barkai, "Iterative Signature Algorithm for Analysis of Large Scale Gene Expression Data. Phys Rev E Stat Nonlin Soft Matter Phys, 67(3 pt 1): 03190201.

[5]- R. Sharan, A. Maron-Katz, N. Arbili, R. Shamir, "CLICK and EXPANDER: a System for Clustering and Visualizing Gene Expression Data", Bioinformatics, 2003.

[6]- Y. Kluger, R. Barsi, JT. Cheng, M. Gerstein, "Spectral Biclustering of Microarray Data: Coclustering Genes and Conditions", Genome Res., 13(4): 703-16, 2003.

[7]- L. Lazzeroni , A. Owen, "Plaid Models for Gene Expression Data", Statistica Sinica, 12: 61-86, 2002

[8]- A. Tanay, R. Sharan, and R. Shamir, "Discovering Statistically Significant Biclusters in Gene Expression Data," *Bioinformatics*, vol. 18, pp. S136-S144, 2002

[9]- J. Yang, H. Wang, W. Wang, and P.S. Yu, "Enhanced Biclustering on Expression Data," *Proc. Third IEEE Conf. Bioinformatics and Bioeng,* pp. 321-327, 2003

[10]- O. Alter, P.O. Brown, D. Botstein, "Processing and Modeling Gene Expression Data Using Singular Decomposition", Proceedings SPIE, vol. 4266 (2001), 171-186

[11]- O. Troyanskaya, M. Cantor, G. Sherlock, P. Brown, T. Hastie, R. Tibshirani, D. Botstein and R. Altman, "Missing Value Estimation for DNA Microarrays", Bioinformatics 17(2001), 1-6.

[12]- A.H. Tewfik, A.B. Tchagang, "Biclustering of DNA Microarray Data with Early Prunning" In Proc, ICASSP 2005.

[13]- S. Tavazoie, J. Hughes, M. Campbell, R. Cho, and G. Church. Yeast micro data set. At http://arep.med.harva*rd.edu/biclustering*

[14]- A. B. Tchagang, A. H Tewfik "Robust Biclustering Algorithm: *ROBA*", Technical Report, University of Minnesota, 2005.

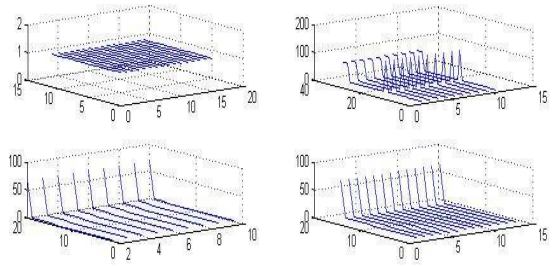

Figure 1: Example of biclusters with constant values, biclusters with constant values on rows, and biclusters with constant values on columns. The x axis represents the conditions, the y axis the genes and z axis the expression level

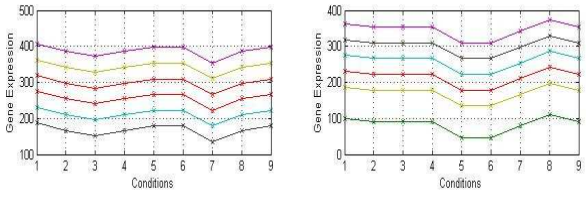

Figure 2: Example of biclusters with coherent values. The lines represent different genes